

REMARKS

Claims 1-34 were presented for examination. Claims 1, 2, 5, 7, 9, 10, 12, 13, 16-19, 23, 24, and 27-37 stand rejected. Claims 3, 4, 6, 8, 11, 14, 15, 20-22, 25, and 26 have been objected to as allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims. Thus, claims 1-34 are presently pending in this application, of which claims 1, 12, and 23 are independent. Applicants submit that claims 1-34 are in condition for allowance. The following comments address all stated grounds of rejection. Applicants respectfully urge the Examiner to pass the claims to allowance in view of the remarks set forth below.

Applicants kindly request the Examiner to expedite prosecution of this application. Applicants note to the Examiner that this is the Applicants' fifth Response to an Office communication addressing the Examiner's art rejections, and in particular, the Cantin and NEC prior art references discussed below. Additionally, Applicants previously filed claim amendments that were not directed to any art rejection but were filed to clarify Applicants' claimed invention to further prosecution. Thus, Applicants respectfully request an interview with the Examiner to discuss any art rejection prior to any further Office Action in order to expedite the prosecution of this application.

Claim Rejections Under 35 U.S.C. §103**I. Claims 1, 7, 12, 18, and 35-36 Stand Rejected Under 35 U.S.C. §103**

Claims 1, 7, 12, 18, and 35-36 stand rejected under 35 U.S.C. §103 as unpatentable over Travis Jr. et al (U.S. Patent No. 5,341,478) ("Travis") in view of Japanese patent application

assigned to NEC Corp., Japanese Patent Application No. 1997JP-0303475 (“NEC”). Applicants respectfully traverse this rejection.

For ease of discussions, summaries of the claimed invention, the primary reference of Travis, and secondary reference of NEC are provided below.

A. Summary of Claimed Invention

The claimed invention is directed towards determining what method of an object to call in an object-oriented environment from a technical computing environment by calculating a ranking of method signatures corresponding to the method. Within a class in an object-oriented environment, each method having the same name must have a different number of inputs, or one or more inputs must differ by data type. Since a method signature represents the number and types of inputs to a method, each method having the same name within that class will have a unique method signature. As such, method signatures can be used to uniquely identify the method that should be called when there are multiple methods with the same name that could be called.

In a technical computing environment, most of the data types are represented as arrays of multiple dimensions. Array-based data types do not distinguish between a scalar, vector or a matrix data type. Because the technical computing environment uses array-based data, it is difficult to invoke methods of objects in an object oriented environment that have the same name and are only distinguished by the data types of their input parameters. For example, the technical computing environment may have its own data type that is not available in the object-oriented environment. As such, an input parameter received from a technical computing environment for

a method call on an object may not fit agreeably into a data type of the object-oriented environment.

In order to determine an appropriate method to call, the claimed invention compares the technical computing environment data to be provided as input to the method with the specified data types expected as described in the method signature. The method signatures are ranked based on which corresponding methods are better suited to accept the input parameters of the data from the calling array-based computing environment. Based on the comparison, the claimed invention automatically selects a method signature according to the ranking and then invokes the method corresponding to the selected method signature.

B. Summary of Travis

Travis is directed towards a method and apparatus for applications to remotely invoke other applications by sending messages with parameters. Travis describes an Applicant Component Architecture Service (ACAS) software component running on both a client and server platform to implement the approach of the invention. Using the name of the message, the ACAS selects from a database a reference to a specific method associated with the message. The ACAS also uses a class database and a context object database to determine and locate the server platform on which to execute the method associated with the message (col. 24, lines 5-17).

The class database includes class and method objects. The class object corresponds to a generic external representation for all members of the set of instances that have the characteristics of the class. The class characteristics are represented by a corresponding set of attributes. For example, a class attribute may include the name of an icon to represent a class on display (see col. 10, line 63 to col. 11, line 11). Each class in the class database also supports a

set of messages. The messages represent operations that each of the instances represented by the corresponding class can support (col. 11, lines 12-37). A message consists of a message name or “verb” and includes parameters having a name, a type, and a direction. The direction indicates if the parameter is an input or output parameter.

A method is derived from application definitions of an application and refers to a certain type of operation or command that may be performed by an application (col. 6, line 66 to col. 7, line 6). A method object identifies an application operation and is associated with method attributes. Unlike the class attributes classes, the method attributes are used to locate and execute an instance associated with a particular method object (col. 12, lines 5-10). For example, the method attributes include a PlatformType, an Interaction Type, and a ServerStartupType attribute (col. 12, lines 14-21). The relationship between class objects and method objects in the class database are provided by method maps. A method map contains a reference to method objects in the class database corresponding to the name of a method object associated with a message (see Fig. 5).

A client application generates a method invocation request message received by the client-side ACAS. The message includes an instance handle, a message with a name and parameter list, a context object handle, and optionally, an output instance handle (see col. 23, lines 49-54). The instance handle is a mechanism to identify the client application’s instance. An instance is derived from application data, and includes items that may be manipulated or accessed by the application. The context object handle is a reference to identify the context object database to be used in the method invocation. After receiving the method invocation request, the context object and class databases are queried to find a method identifier (see Fig. 6). Entries, if any, in the context object database are compared with the attributes in the set of

methods objects referenced by the method map to select the method object and the appropriate method to execute the desired operation represented by the message (see Figure 16 and col. 26, lines 54-59). For example, a method object with an attribute indicating a PlatformType of A may be selected for the method invocation request. This indicates the operation associated with the method object should be performed on a server platform of a specific type.

Upon determining the appropriate method, the client-side ACAS queries a server registration facility and the context object database to find a method server on the desired server platform on which to execute the method associated with the method identifier. The ACAS packages a method invocation request and transmits to the ACAS component corresponding to the method server of the desired server platform. The server-side ACAS unpackages the method invocation request into a data structure recognizable by the server platform (col. 32, lines 30-44) and dispatches the method server to execute the operation (col. 32, lines 62-66).

C. Summary of NEC

NEC is directed toward generating a database index key comprising a composite index key to improve search methods for objects in an object-oriented database management system. A database index is used to locate data in a database. A database index allows a search of the database for a specific instance of data, e.g. a row of data having a specific key value, without searching the entire database, much like using an index for an instruction manual. A search looks at the index to find the location of the data and then obtains the data from that location. Database indexing is used to improve database searching performance.

NEC describes a method for creating a composite index key from a data structure stored in the database representing property values of an object. The composite key index then

becomes the basis of a management object subsequently used to retrieve objects. For example, a data structure for employee information may contain two variables representing the name of the employee and affiliation coding. These variables represent the properties of an object stored in the tables of the database. For indexing purposes, the variables of the data structure comprising the composite index key are ranked by size. The size would indicate the size of the data the variables can hold.

D. Independent Claims 1 and 12 Stand Rejected Under 35 U.S.C. §103

Independent claims 1 and 12 stand rejected under 35 U.S.C. §103(a) as unpatentable over Travis in view of NEC. Claims 1 and 12 are independent claims. Applicants respectfully traverse this rejection.

Claims 1 and 12 are directed to a method and computer-readable medium claim, respectively. These claims recite a method for retrieving a set of method signatures for a method referenced in a requested method invocation. Each method signature includes a method name and data types of input parameters to be received by the corresponding method. The method includes *comparing the data types of input parameters of each method represented by signatures to data types of input parameters passed by the requested method invocation to determine suitability of each method to receive input parameters passed by the requested method invocation.* These claims further recite ranking the method signatures as a function of the comparison, selecting one of the method signatures according to the ranking, and invoking the method corresponding to the selected method signature. Travis in view of NEC does not teach or suggest each and every feature recited in these claims.

Travis does not teach or suggest *comparing the data types of input parameters of each method represented by signatures to data types of input parameters passed by the requested method invocation* to determine suitability of each method to receive input parameters passed by the requested method invocation. Rather, Travis describes using a method map to associate a name of a message to a method object in a class database. Instead of comparing data types of input parameters of methods represented by signatures to data types of input parameters passed by the requested method invocation as in the claimed invention, Travis compares entries in the context object database with the attributes in the set of method objects referenced by the method map. The method attributes of Travis are used to locate a particular application instance and server platform, and are not passed to the invoked method. Thus, Travis does not discuss comparing data types of input parameters received by the requested method invocation to the data types to be passed into the invoked method.

Furthermore, Travis does not teach or suggest comparing the data types of input parameters of each method represented by signatures to data types of input parameters passed by the requested method invocation *to determine suitability of each method to receive input parameters passed by the requested method invocation*. Rather, Travis compares database entries to determine the method to use on a desired server platform. Travis does not discuss determining the suitability of each method to receive input parameters passed by the requested method invocation. In Travis, parameters are passed via a message associated with a method as determined by the ACAS. However, Travis does not determine the suitability of the parameters to be received by the method to be invoked. Instead, Travis calls the method corresponding to the desired server platform from the method map of the message. Therefore, Travis fails to teach or suggest comparing the data types of input parameters of each method represented by

signatures to data types of input parameters passed by the requested method invocation *to determine suitability of each method to receive input parameters passed by the requested method invocation.*

Examiner admits in the Office Action that Travis does not teach or suggest the recited claim limitations of ranking the method signatures as a function of the comparison, and selecting one of the method signatures according to the ranking. The Examiner cites NEC for the purpose of suggesting one ordinarily skilled in the art might modify Travis to rank method signatures as a function of the comparison, and selecting one of the method signatures according to the ranking. NEC does not teach or suggest *comparing the data types of input parameters of each method represented by signatures to data types of input parameters passed by the requested method invocation to determine suitability of each method to receive input parameters passed by the requested method invocation.* Instead, NEC discusses generating a database index key from variables in a database. As discussed above, Travis fails to teach or suggest comparing the data types of input parameters of each method represented by signatures to data types of input parameters passed by the requested method invocation *to determine suitability of each method to receive input parameters passed by the requested method invocation.* Therefore, NEC fails to bridge the factual deficiencies of Travis with regards to these claim limitations.

For at least the aforementioned reasons, Applicants submit that Travis in view of NEC does not detract from the patentability of independent claims 1 and 12. Claims 7 and 35 depend on and incorporate the patentable subject matter of independent claim 1, and claims 18 and 36 depend on and incorporate the patentable subject matter of independent claim 12. As such, Applicants submit that Travis in view of NEC does not detract from the patentability of

dependent claims 7, 18, and 35-36. Accordingly, Applicants respectfully request the Examiner to withdraw the rejection of claims 1, 7, 12, 18, and 35-36 under 35 U.S.C. §103.

II. Claims 23 and 37 Stand Rejected Under 35 U.S.C. §103

Claims 23 and 37 stand rejected under 35 U.S.C. §103 as unpatentable over Cantin et al (EP 0 690 375 A2) (“Cantin”) in view of NEC and in further view of Travis.

A. Independent Claim 23 Stands Rejected Under 35 U.S.C. §103

Independent claim 23 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Cantin in view of NEC and in further view of Travis. Applicants respectfully traverse this rejection.

Claim 23 recites a system comprising an object-oriented environment and a technical computing environment. The object-oriented environment includes an interface for identifying methods provided by objects. The technical computing environment comprises a calculation workspace, a command interpreter, and a signature selector. When the calculation workspace encounters a requested method invocation, the signature selector retrieves and ranks a list of signatures corresponding to the method referenced in the requested method invocation. The command interpreter invokes in the object-oriented environment one of the methods represented by one of the signatures as a function of the ranking. The ranking *determines suitability of data types of input parameters of each method represented by the signatures to receive data types of input parameters passed by the requested method invocation.* Cantin in view of NEC in further view of Travis does not teach or suggest each and every feature recited in claim 23.

Cantin does not teach or suggest ranking the signature to *determine suitability of data types of input parameters of each method represented by the signatures to receive data types of input parameters passed by the requested method invocation*. Rather, Cantin calls specialized methods to store objects to a database table. For each value of a property of the object to be stored to the database, the specialized method writes the property values to the database table. As such, Cantin is focused on storing properties of an object to a database table. In contrast to the claimed invention, Cantin does not discuss ranking signatures to *determine suitability of data types of input parameters of each method represented by the signatures to receive data types of input parameters passed by the requested method invocation*. Therefore, Cantin fails to teach or suggest ranking the signature to *determine suitability of data types of input parameters of each method represented by the signatures to receive data types of input parameters passed by the requested method invocation*.

The Examiner cites NEC for the purpose of suggesting one ordinarily skilled in the art might modify Cantin to provide a signature selector to rank a list of signatures according to the claimed invention. NEC discusses ranking the size of variables of data structures stored in the database for using as a composite index key for searching the database. As a database, NEC uses the composite index to search the database to retrieve records from a database. NEC does not teach or suggest a signature selector retrieving a list of method signatures *and ranking the signatures to determine suitability of data types of input parameters of each method represented by the signatures to receive data types of input parameters passed by the requested method invocation*. As such, NEC fails to bridge the factual deficiencies of Cantin.

The Examiner cites Travis for the purpose of suggesting one ordinarily skilled in the art might modify Cantin to provide for input parameters passed by a requested method invocation.

Although Travis may describe input parameters from a message corresponding to a method, Travis does not teach or suggest ranking the signatures to *determine suitability of data types of input parameters of each method represented by the signatures to receive data types of input parameters passed by the requested method invocation*. As such, Travis fails to bridge the factual deficiencies of Cantin.

For at least the aforementioned reasons, Applicants submit that Cantin in view of NEC and in further view of Travis does not detract from the patentability of independent claim 23. Claims 24-34 depend on and incorporate the patentable subject matter of independent claim 23. As such, Applicants submit that Cantin in view of NEC and in further view of Travis does not detract from the patentability of dependent claims 24-34. Accordingly, Applicants respectfully request the Examiner to withdraw the rejection of claims 23-34 under 35 U.S.C. §103.

III. Additional Dependent Claim Rejections Under 35 U.S.C. §103

Dependent claims 2, 13, and 24 are rejected under 35 U.S.C. §103(a) as unpatentable over Travis in view of NEC and in further view of Admitted Prior Art.

Dependent claim 5 is rejected under 35 U.S.C. §103(a) as unpatentable over Travis in view of NEC and in further view of Hartmut Pohlheim (“*Genetic and Evolutionary Algorithm Toolbox for use with MATLAB*”).

Dependent claims 9, 16-17, and 19 are rejected under 35 U.S.C. §103(a) as unpatentable over Travis in view of NEC and in further view of Cantin.

Dependent claim 10 is rejected under 35 U.S.C. §103(a) as unpatentable over Travis in view of NEC and Travis and in further view of Bill Venners (“Eternal Math”).

Dependent claims 27-29 and 34 are rejected under 35 U.S.C. §103(a) as unpatentable over Cantin in view of NEC and Travis and in further view of Hartmut Pohlhei.

Dependent claims 31-33 are rejected under 35 U.S.C. §103(a) as unpatentable over Cantin in view of NEC and Travis and in further view of Bill Vanners

Dependent claim 30 is rejected under 35 U.S.C. §103(a) as unpatentable over Cantin in view of NEC and Travis and in further view of John W. Eaton (“A High Level Interactive Language for Numerical Computations, Edition 3 for Octave Version 2.1x”).

None of the cited references, alone or in combination, disclose, teach, or suggest each and every feature of independent claims 1, 12, and 23 as discussed above. Claims 2, 5, 9, and 10 depend on an incorporate the patentable subject matter of independent claim 1. Claims 13, 16-17, and 19 depend on an incorporate the patentable subject matter of independent claim 12. Claims 24, 27-30, and 34 depend on an incorporate the patentable subject matter of independent claim 23. As such, Applicants submit dependent claims 2, 5, 9, 10, 13, 16-17, 19, 24, 27-30, and 34 are patentable and in condition for allowance. Accordingly, Applicants respectfully request the Examiner to withdraw the rejection of claims 2, 5, 9, 10, 13, 16-17, 19, 24, 27-30, and 34 under 35 U.S.C. §103.

CONCLUSION

In light of the aforementioned arguments, Applicants contend that each of the Examiners rejections has been adequately addressed and the pending application is in condition for allowance.

In order to expedite prosecution of this application, Applicants respectfully request the Examiner to contact the Applicants' attorney at the telephone number identified below.

Respectfully submitted,

LAHIVE & COCKFIELD, LLP



Christopher J. McKenna
Registration No. 53,302
Attorney for Applicants

Lahive & Cockfield, LLP
28 State Street
Boston, MA 02109
(617) 227-7400

Dated: April 25, 2005